M 11

```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
                                                G  1
**FILE**ID**EXPRESS




EEEEEEEEEE  XX      XX  PPPPPPPP    RRRRRRRR    EEEEEEEEEE    SSSSSSSS    SSSSSSSS
EEEEEEEEEE  XX      XX  PPPPPPPP    RRRRRRRR    EEEEEEEEEE    SSSSSSSS    SSSSSSSS
EE          XX      XX  PP      PP  RR      RR  EE          SS          SS
EE          XX      XX  PP      PP  RR      RR  EE          SS          SS
EE            XX  XX    PP      PP  RR      RR  EE          SS          SS
EE            XX  XX    PP      PP  RR      RR  EE          SS          SS
EEEEEEEE        XX      PPPPPPPP    RRRRRRRR    EEEEEEEE      SSSSSS      SSSSSS
EEEEEEEE        XX      PPPPPPPP    RRRRRRRR    EEEEEEEE      SSSSSS      SSSSSS
EE            XX  XX    PP          RR  RR      EE                  SS          SS
EE            XX  XX    PP          RR  RR      EE                  SS          SS
EE          XX      XX  PP          RR    RR    EE                  SS          SS      ....
EE          XX      XX  PP          RR    RR    EE                  SS          SS      ....
EEEEEEEEEE  XX      XX  PP          RR      RR  EEEEEEEEEE  SSSSSSSS    SSSSSSSS        ....
EEEEEEEEEE  XX      XX  PP          RR      RR  EEEEEEEEEE  SSSSSSSS    SSSSSSSS        ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000          .TITLE  EXPRESS - EXPRESSION ANALYSIS
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ; EXPRESSION ANALYSIS
0000    ;
0000    ; D. N. CUTLER 9-MAY-77
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-003 PCG0007         Peter George    27-Jul-1983
0000    ;               Add dummy LIB$SCOPY_DXDX routine.
0000    ;
0000    ;       V03-002 PCG0006         Peter George    12-Jul-1983
0000    ;               Fix bug in buffer overflow calculation.
0000    ;
0000    ;       V03-001 PCG0005         Peter George    18-Nov-1982
0000    ;               Upcase symbol names when evaluating F$VERIFY arguments
0000    ;               in a comment.
0000    ;
0000    ;---
```

```
0000     ;
0000     ; MACRO LIBRARY CALLS
0000     ;
0000
0000             PRCDEF                                  ;DEFINE PROCESS WORK AREA
0000             WRKDEF                                  ;DEFINE COMMAND WORK AREA
0000             $CLIMSGDEF                              ;DEFINE ERROR/STATUS CODES
0000
0000     ;
0000     ; LOCAL MACROS
0000     ;
0000     ; GENERATE OPERATOR/OPERAND TABLE AND SYMBOL NAMES
0000     ;
0000
0000             .MACRO  GENOP NAME,INDEX,TYPE,PREC,CHAR,?L1,?L2,?L3,?L4
0000     OPI_K_'NAME=INDEX
0000             .IF IDN <TYPE>,<OPERATOR>
0000     OPP_K_'NAME=PREC
0000             .IF DIF <NAME>,<SOS>
0000             .IF DIF <NAME>,<EOS>
0000             .IF DIF <NAME>,<STORE>
0000     L1:     .BYTE   L4-L1
0000             .BYTE   L3-L2
0000             .IF NB  <CHAR>
0000     L2:     .ASCII  \CHAR\
0000             .IFF
0000     L2:     .ASCII  /NAME/
0000             .ENDC
0000     L3:     .BYTE   OPP_K_'NAME
0000     L4:     .BYTE   OPI_K_'NAME
0000             .ENDC
0000             .ENDC
0000             .ENDC
0000             .ENDC
0000             .ENDM   GENOP
0000
0000     ;
0000     ; LOCAL SYMBOLS
0000     ;
0000     ; DEFINE STACK ITEM OFFSETS
0000     ;
0000
0000             $DEFINI STK
0000
0000     $DEF    STK_W_TYPE      .BLKW   1       ;STACK ITEM TYPE
0002     $DEF    STK_W_PREC                      ;OPERATOR PRECEDENCE
0002     $DEF    STK_W_SIZE      .BLKW   1       ;SIZE OF OPERAND VALUE IN BYTES
0004     $DEF    STK_L_ADDR      .BLKL   1       ;ADDRESS OF OPERAND VALUE
0008     $DEF    STK_K_LENGTH                    ;LENGTH OF STACK ITEM
0008
0008             $DEFEND STK
0000
0000     ;
0000     ; LOCAL DATA
0000     ;
0000     ; OPERATOR TABLE
0000     ;
```

```
          0000
      00000000          .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
          0000
          0000  OPTAB:  GENOP   SOS,30,OPERATOR,0           ;START OF STATEMENT
          0000          GENOP   EOS,32,OPERATOR,1           ;END OF STATEMENT
          0000          GENOP   LPAREN,34,OPERATOR,2,<(>    ;LEFT PARENTHESIS
          0005          GENOP   RPAREN,36,OPERATOR,3,<)>    ;RIGHT PARENTHESIS
          000A          GENOP   STORE,38,OPERATOR,0         ;STORE RESULT
          000A          GENOP   AND,0,OPERATOR,5            ;BOOLEAN AND
          0011          GENOP   NOT,2,OPERATOR,6            ;BOOLEAN NOT
          0018          GENOP   OR,4,OPERATOR,4             ;BOOLEAN OR
          001E          GENOP   ADD,6,OPERATOR,8,<+>        ;INTEGER ADD
          0023          GENOP   SUB,8,OPERATOR,8,<->        ;INTEGER SUBTRACT
          0028          GENOP   MUL,10,OPERATOR,9,<*>       ;INTEGER MULTIPLY
          002D          GENOP   DIV,12,OPERATOR,9,</>       ;INTEGER DIVIDE
          0032          GENOP   NEG,14,SPECIAL              ;INTEGER NEGATE
          0032          GENOP   POS,16,SPECIAL              ;INTEGER NOOP
0000000A  0032  OPP_K_NEG=OPP_K_DIV+1                       ;ONE GREATER THAN HIGEST ARITHMETIC
0000000A  0032  OPP_K_POS=OPP_K_DIV+1                       ;ONE GREATER THAN HIGEST ARITHMETIC
          0032          GENOP   EQ,18,OPERATOR,7            ;ARITHMETIC EQUAL
          0038          GENOP   GE,20,OPERATOR,7            ;ARITHMETIC GREATER OR EQUAL
          003E          GENOP   GT,22,OPERATOR,7            ;ARITHMETIC GREATER
          0044          GENOP   LE,24,OPERATOR,7            ;ARITHMETIC LESS OR EQUAL
          004A          GENOP   LT,26,OPERATOR,7            ;ARITHMETIC LESS
          0050          GENOP   NE,28,OPERATOR,7            ;ARITHMETIC NOT EQUAL
          0056          GENOP   EQS,40,OPERATOR,7           ;STRING EQUAL
          005D          GENOP   GES,42,OPERATOR,7           ;STRING GREATER OR EQUAL
          0064          GENOP   GTS,44,OPERATOR,7           ;STRING GREATER
          006B          GENOP   LES,46,OPERATOR,7           ;STRING LESS OR EQUAL
          0072          GENOP   LTS,48,OPERATOR,7           ;STRING LESS
          0079          GENOP   NES,50,OPERATOR,7           ;STRING NOT EQUAL
00000034  0080  OPI_K_OPERAND = 52                          ;LOWEST OPERAND INDEX
          0080          GENOP   STRING,52,OPERAND           ;UNEVALUATED CHARACTER STRING OPERAND
          0080          GENOP   STACK,54,OPERAND            ;EVALUATED ON-STACK OPERAND
        00  0080        .BYTE   0                           ;TABLE TERMINATOR BYTE
          0081
          0081  ;
          0081  ; SIZE OF EXPRESSION ANALYSIS STACK
          0081  ;
          0081
00000084  0081  PARSESTKSIZ = 132                           ;SIZE OF PARSE STACK
0000016C  0081  TRIADSTKSIZ = 364                           ;SIZE OF TRIAD STACK
          0081
          0081  ;
          0081  ; LOCAL VARIABLES ON STACK
          0081  ;
          0081
          0081                  $DEFINI
00000001  0000  NESTLVL:        .BLKB   1                   ; CURRENT LEVEL OF NESTING
00000002  0001  REQMODE:        .BLKB   1                   ; REQUIRED EXPRESSION MODE
          0002                                              ; (0=CAN BE ANY MODE)
00000003  0002  CURMODE:        .BLKB   1                   ; CURRENT EXPRESSION MODE
00000004  0003                  .BLKB   1
0000000C  0004  RESULT:         .BLKQ   1                   ; EXPRESSION RESULT
          000C  LOCALSIZ:
          000C                  $DEFEND
          0081
```

```
                   0081  ;
                   0081  ;  'TRUE' AND 'FALSE'  VALUES
                   0081  ;
                   0081
      45 55 52 54  0081  TRUE:    .ASCII  'TRUE'               ;
   45 53 4C 41 46  0085  FALSE:   .ASCII  'FALSE'              ;
      79 59 74 54  008A  TRUSYM:  .ASCII  'TtYy'               ; NOTE-ORDER IS ASSUMED ELSEWHERE
                   008E
```

```
           008E         .SBTTL  DUMMY LIB$SCOPY_DXDX ROUTINE
           008E
     0000  008E         .ENTRY  LIB$SCOPY_DXDX,0
       04  0090         RET
```

```
                0091            .SBTTL  EVALUATE BINARY EXPRESSION
                0091      ;+
                0091      ; DCL$BINEXPR - EVALUATE BINARY EXPRESSION
                0091      ;
                0091      ; THIS ROUTINE IS CALLED TO ANALYZE AN EXPRESSION AND RETURN AN ARITHMETIC RESULT.
                0091      ;
                0091      ; INPUTS:
                0091      ;
                0091      ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                0091      ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                0091      ;
                0091      ; OUTPUTS:
                0091      ;
                0091      ;       R0 = STATUS
                0091      ;       R1 = BINARY LONGWORD VALUE
                0091      ;
                0091      ;       R2,R3 DESTROYED.
                0091      ;-
                0091
                0091      DCL$BINEXPR::
      2D    10  0091            BSBB    DCL$EXPRESS             ;EVALUATE EXPRESSION
   01 50    E8  0093            BLBS    R0,DCL$CVT_BINARY       ;IF OK, CONVERT RESULT TO BINARY
            05  0096            RSB
```

```
                        0097          .SBTTL  CONVERT STRING OPERAND TO BINARY
                        0097     ;+
                        0097     ; DCL$CVT_BINARY - CONVERT STRING OPERAND TO BINARY
                        0097     ;
                        0097     ; THIS ROUTINE IS CALLED TO CONVERT A EXPRESSION RESULT TO BINARY.
                        0097     ; IF THE RESULT IS ALREADY BINARY, THEN THAT VALUE IS RETURNED.
                        0097     ;
                        0097     ; INPUTS:
                        0097     ;
                        0097     ;        R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0097     ;        R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        0097     ;
                        0097     ;        R1/R2 = QUADWORD DESCRIBING EXPRESSION VALUE:
                        0097     ;                IF R2 NONZERO, R1/R2 ARE A STRING DESCRIPTOR
                        0097     ;                IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                        0097     ;
                        0097     ; OUTPUTS:
                        0097     ;
                        0097     ;        R0 = STATUS
                        0097     ;        R1 = BINARY LONGWORD VALUE
                        0097     ;        R2 = 0 (TO INDICATE RESULT IS BINARY)
                        0097     ;
                        0097     ;        R3 DESTROYED.
                        0097     ;-
                        0097
                        0097     DCL$CVT_BINARY::
               52   D5  0097              TSTL    R2                      ;STRING OR BINARY VALUE?
               1F   13  0099              BEQL    80$                     ;OK IF BINARY VALUE
               52   DD  009B              PUSHL   R2                      ;SAVE STARTING ADDRESS OF STRING
          52   51   7D  009D              MOVQ    R1,R2                   ;COPY STRING DESCRIPTOR
     51 00AE   CB   9A  00A0              MOVZBL  PRC_B_DEFRADIX(R11),R1  ;SET RADIX FOR CONVERSION
            FF58'   30  00A5              BSBW    DCL$CNVASCBIN           ;CONVERT TO BINARY
               04   BA  00A8              POPR    #^M<R2>                 ;RESTORE STARTING ADDRESS OF STRING
               0E   13  00AA              BEQL    80$                     ;IF SUCCESSFUL, EXIT WITH R1
                        00AC     ;
                        00AC     ; INVALID CHARACTER IN STRING-CHECK FOR BEGINNING LETTER "TtYy"
                        00AC     ; IF "T", "t", "Y", OR "y", THEN RETURN 1, ELSE RETURN 0.
                        00AC     ;
D9 AF     04   62   3A  00AC              LOCC    (R2),#4,TRUSYM          ;CHECK FOR TRUE CHARACTERS
               05   13  00B1              BEQL    70$                     ;BRANCH IF NONE FOUND
          51   01   D0  00B3              MOVL    #1,R1                   ;SET RESULT TO TRUE
               02   11  00B6              BRB     80$                     ;SKIP
               51   D4  00B8     70$:     CLRL    R1                      ;SET RESULT TO FALSE
          50   01   D0  00BA     80$:     MOVL    #1,R0                   ;SUCCESSFUL
               52   D4  00BD              CLRL    R2                      ;INDICATE RESULT IS BINARY
                    05  00BF              RSB
```

```
                     00C0          .SBTTL  EXPRESSION ANALYSIS
                     00C0    ;+
                     00C0    ; DCL$EXPRESS - EXPRESSION ANALYSIS
                     00C0    ;
                     00C0    ; THIS ROUTINE IS CALLED TO ANALYZE AN EXPRESSION AND RETURN A BINARY
                     00C0    ; OR CHARACTER STRING RESULTANT VALUE.
                     00C0    ;
                     00C0    ; INPUTS:
                     00C0    ;
                     00C0    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                     00C0    ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                     00C0    ;
                     00C0    ; OUTPUTS:
                     00C0    ;
                     00C0    ;       R0 = STATUS
                     00C0    ;       R1/R2 = QUADWORD DESCRIBING EXPRESSION VALUE:
                     00C0    ;               IF R2 NONZERO, R1/R2 ARE A STRING DESCRIPTOR
                     00C0    ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                     00C0    ;
                     00C0    ;       R3 DESTROYED.
                     00C0    ;-
                     00C0
                     00C0   DCL$EXPRESS::                                   ;ANALYZE EXPRESSION
      51   01   90   00C0           MOVB    #PRC_K_DEC,R1                   ;SET TO DEFAULT DECIMAL RADIX
                     00C3   DCL$EXPRADIX::                                  ;ALTERNATE ENTRY-RADIX SET EXTERNALLY
    13F0   8F   BB   00C3           PUSHR   #^M<R4,R5,R6,R7,R8,R9,AP> ;SAVE REGISTERS
    F486   CA   DD   00C7           PUSHL   WRK_L_EXPANDPTR(R10)    ;SAVE CURRENT PARSE POSITION
 00AE CB   51   90   00CB           MOVB    R1,PRC_B_DEFRADIX(R11)  ;SET THE RADIX FOR LATER
   F0 AA   20   A8   00D0           BISW    #WRK_M_STAR,WRK_W_FLAGS(R10) ;SET ASTERISK TERMINATOR FLAG
                     00D4
      5E   0C   C2   00D4           SUBL    #LOCALSIZ,SP                    ;ALLOCATE SCRATCH SPACE ON STACK
      57   5E   D0   00D7           MOVL    SP,R7                           ;POINT TO SCRATCH SPACE
           67   94   00DA           CLRB    NESTLVL(R7)                     ;INITIALIZE PARENTHESIS NESTING LEVEL
   01 A7   53   90   00DC           MOVB    R3,REQMODE(R7)                  ;SET EXPECTED EXPRESSION MODE
                     00E0
      58   5E   D0   00E0           MOVL    SP,R8                           ;SET ADDRESS OF PARSE STACK
5E  FE10   CE   9E   00E3           MOVAB   -<TRIADSTKSIZ+PARSESTKSIZ>(SP),SP ;ALLOCATE SPACE FOR STACKS
      59   5E   D0   00E8           MOVL    SP,R9                           ;SET BASE ADDRESS OF TRIAD STACK
      78   1E   D0   00EB           MOVL    #<OPP_K_SOS>@16+OPI_K_SOS,-(R8) ;INITIALIZE PARSE STACK
      78   68   7D   00EE           MOVQ    (R8),-(R8)                      ;DUPLICATE FIRST ITEM FOR ERROR CHECK
                     00F1
                     00F1    ;
                     00F1    ; PARSE NEXT ITEM
                     00F1    ;
                     00F1
        FF0C'   30   00F1   10$:    BSBW    DCL$SETCHAR                     ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
      50   20   91   00F4           CMPB    #^A/ /,R0                       ;BLANK?
           03   12   00F7           BNEQ    20$                             ;IF NEQ NO
        FF04'   30   00F9           BSBW    DCL$MOVCHAR                     ;MOVE CHARACTER TO COMMAND BUFFER
        FF01'   30   00FC   20$:    BSBW    DCL$MARK                        ;MARK CURRENT PARSE POSITION
        FEFE'   30   00FF           BSBW    DCL$SETCHAR                     ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
      50   2E   91   0102           CMPB    #^A/./,R0                       ;OPERATOR?
           21   13   0105           BEQL    25$                             ;IF NEQ NO
        FEF6'   30   0107           BSBW    DCL$GETOKEN                     ;GET NEXT TOKEN FROM INPUT BUFFER
           30   13   010A           BEQL    40$                             ;IF EQL NONE
      34   68   B1   010C           CMPW    STK_W_TYPE(R8),#OPI_K_OPERAND ;TOS AN OPERAND?
           0E   19   010F           BLSS    80$                             ;BR IF OPERATOR IS TOP
```

```
               51    04   D1  0111    302           CMPL    #4,R1                      ;POSSIBLY 'THEN' KEY WORD?
                     09   12  0114    303           BNEQ    80$                        ;IF NEQ NO
      62  4E454854 8F   D1  0116    304           CMPL    #^A/THEN/,(R2)             ;'THEN' KEYWORD?
                     3C   13  011D    305           BEQL    130$                       ;IF SO, TERMINATE EXPRESSION PARSE
                              011F    306
                              011F    307 ;
                              011F    308 ; PARSE OPERAND TOKEN
                              011F    309 ;
                              011F    310
                   015A   30  011F    311 80$:      BSBW    PARSE_OPERAND              ;PARSE THE OPERAND
                              0122    312                                              ;(RESULTS MAY BE PLACED ON STACK)
                  14 50   E9  0122    313           BLBC    R0,35$                     ;EXIT IF ERROR DETECTED
                   00A6   31  0125    314           BRW     170$                       ;STACK THE OPERAND PARAMETERS
                              0128    315
                              0128    316 ;
                              0128    317 ; PARSE .XX. OPERATORS
                              0128    318 ;
                              0128    319
                  FED5'   30  0128    320 25$:      BSBW    DCL$MOVBTOKN               ;MOVE TERMINATOR AND GET NEXT BLANK TOKEN
               50    2E   91  012B    321           CMPB    #^A/./,R0                  ;TERMINATOR PERIOD?
                     03   12  012E    322           BNEQ    30$                        ;IF NEQ NO
                  FECD'   30  0130    323           BSBW    DCL$MOVCHAR                ;MOVE TERMINATOR TO EXPANSION BUFFER
                   0114   30  0133    324 30$:      BSBW    FIND_OPERATOR              ;SEARCH OPERATOR TABLE
                  2A 50   E8  0136    325           BLBS    R0,50$                     ;IF FOUND, PROCESS OPERATOR
                   02AA   31  0139    326 35$:      BRW     EXIT                       ;EXIT WITH ERROR
                              013C    327
                              013C    328 ;
                              013C    329 ; PARSE OPERATOR CHARACTER
                              013C    330 ;
                              013C    331 ; TERMINATE EXPRESSION IF END-OF-LINE OR RIGHT PAREN W/O CORRESP. LEFT
                              013C    332 ;
                              013C    333
          52    F486 CA   D0  013C    334 40$:      MOVL    WRK_L_EXPANDPTR(R10),R2 ;SET ADDRESS OF START OF SYMBOL
               29    50   91  0141    335           CMPB    R0,#^A')'                  ;RIGHT PAREN?
                     04   12  0144    336           BNEQ    45$                        ;BRANCH IF NOT
                     67   95  0146    337           TSTB    NESTLVL(R7)                ;IS THERE A CORRESPONDING LEFT PAREN?
                     11   15  0148    338           BLEQ    130$                       ;IF NO LEFT PAREN, TERMINATE EXPRESSION
                  FEB3'   30  014A    339 45$:      BSBW    DCL$MOVCHAR                ;MOVE TERMINATOR TO EXPANSION BUFFER
                     0C   13  014D    340           BEQL    130$                       ;IF EQL END OF LINE
               51    01   D0  014F    341           MOVL    #1,R1                      ;SET LENGTH OF STRING
                   00F5   30  0152    342           BSBW    FIND_OPERATOR              ;SEARCH OPERATOR TABLE
                  0B 50   E8  0155    343           BLBS    R0,50$                     ;IF FOUND, PROCESS THE OPERATOR
                  FEA5'   30  0158    344           BSBW    DCL$BACKUPMOVE             ;BACKUP MOST RECENT MOVCHAR
                              015B    345
                              015B    346 ;
                              015B    347 ; MARK END OF EXPRESSION
                              015B    348 ;
                              015B    349
               51    01   9A  015B    350 130$:     MOVZBL  #OPP_K_EOS,R1              ;SET END OF STATEMENT PRECEDENCE VALUE
               53    20   9A  015E    351           MOVZBL  #OPI_K_EOS,R3              ;SET END OF STATEMENT OPERATOR INDEX
                     36   11  0161    352           BRB     140$                       ;FORCE EVERYTHING TO TRIAD STACK
                              0163    353
                              0163    354 ;
                              0163    355 ; PROCESS OPERATOR
                              0163    356 ;
                              0163    357 ; IF LEFT PARENTHESIS, INCREMENT THE NESTING LEVEL AND MAKE SURE
                              0163    358 ; AN OPERATOR PRECEEDS IT.
```

```
                        0163   359 ;
                        0163   360
       53    22    D1   0163   361 50$:     CMPL     #OPI_K_LPAREN,R3            ;LEFT PARENTHESIS?
             0A    12   0166   362          BNEQ     53$                         ;IF NEQ NO
             67    96   0168   363          INCB     NESTLVL(R7)                 ;INCREMENT PAREN NESTING LEVEL
       34    68    B1   016A   364          CMPW     STK_W_TYPE(R8),#OPI_K_OPERAND ;TOP OF STACK OPERAND?
             5F    19   016D   365          BLSS     170$                        ;IF GTR NO
           00BB    31   016F   366          BRW      250$                        ;REPORT EXPRESSION SYNTAX ERROR
                        0172   367
                        0172   368 ;
                        0172   369 ; CHECK FOR LEGITIMATE UNARY OPERATORS.  CONVERT PLUS AND MINUS TO INTEGER
                        0172   370 ; NOOP AND NEGATE.  PLACE OPERATOR ON STACK.
                        0172   371 ;
                        0172   372
       34    68    B1   0172   373 53$:     CMPW     STK_W_TYPE(R8),#OPI_K_OPERAND    ;TOS AN OPERATOR?
             22    18   0175   374          BGEQ     140$                             ;IF NO
       53    06    B1   0177   375          CMPW     #OPI_K_ADD,R3                    ;THIS A UNARY PLUS?
             08    12   017A   376          BNEQ     57$                              ;IF NEQ NO, SKIP TO NEXT
       53    10    B0   017C   377          MOVW     #OPI_K_POS,R3                    ;CONVERT THIS TO INTEGER NOOP
       51    0A    B0   017F   378          MOVW     #OPP_K_POS,R1                    ;...
             4A    11   0182   379          BRB      170$                             ;GO STACK THIS
       53    08    B1   0184   380 57$:     CMPW     #OPI_K_SUB,R3                    ;THIS A UNARY MINUS?
             08    12   0187   381          BNEQ     63$                              ;IF NEQ NO
       53    0E    B0   0189   382          MOVW     #OPI_K_NEG,R3                    ;CONVERT THIS TO NEGATE
       51    0A    B0   018C   383          MOVW     #OPP_K_NEG,R1                    ;...
             3D    11   018F   384          BRB      170$                             ;GO STACK THIS
       53    02    B1   0191   385 63$:     CMPW     #OPI_K_NOT,R3                    ;NOT LPAREN, ADD OR SUB. ONLY LEGAL
             38    13   0194   386          BEQL     170$                             ;IF .NOT., STACK OPERATOR
           0094    31   0196   387          BRW      250$                             ;GO REPORT SYNTAX ERROR
                        0199   388
                        0199   389 ;
                        0199   390 ; COMPARE OPERATOR PRECEDENCE VALUE WITH OPERATOR IN NEXT TO LAST ITEM ON STACK
                        0199   391 ;
                        0199   392
 34       08 A8    B1   0199   393 140$:    CMPW     STK_W_TYPE+STK_K_LENGTH(R8),#OPI_K_OPERAND ;PREVIOUS ITEM OPERATOR?
             06    18   019D   394          BGEQ     150$                        ;IF LEQ NO
 51       0A A8    B1   019F   395          CMPW     STK_W_PREC+STK_K_LENGTH(R8),R1 ;STACK OPERATOR HIGHER PRECEDENCE?
             3A    18   01A3   396          BGEQ     190$                        ;IF GEQ YES
       53    24    D1   01A5   397 150$:    CMPL     #OPI_K_RPAREN,R3            ;CURRENT OPERATOR RIGHT PARENTHESIS?
             1F    12   01A8   398          BNEQ     160$                        ;IF NEQ NO
             67    97   01AA   399          DECB     NESTLVL(R7)                 ;DECREMENT PAREN NESTING LEVEL
 08 A8       02    B1   01AC   400          CMPW     #OPI_K_NOT,STK_W_TYPE+STK_K_LENGTH(R8)    ;OPERATOR BOOLEAN NOT?
             2D    13   01B0   401          BEQL     190$                                      ;IF EQL YES
 08 A8       0E    B1   01B2   402          CMPW     #OPI_K_NEG,STK_W_TYPE+STK_K_LENGTH(R8)    ;OPERATOR NEGATE?
             27    13   01B6   403          BEQL     190$                                      ;IF EQL YES
 08 A8       10    B1   01B8   404          CMPW     #OPI_K_POS,STK_W_TYPE+STK_K_LENGTH(R8)    ;OPERATOR INTEGER NOOP?
             21    13   01BC   405          BEQL     190$                                      ;IF EQL YES
 08 A8       22    B1   01BE   406          CMPW     #OPI_K_LPAREN,STK_W_TYPE+STK_K_LENGTH(R8) ;OPERATOR LEFT PAREN?
             69    12   01C2   407          BNEQ     250$                        ;IF NEQ NO
       68    88    7D   01C4   408          MOVQ     (R8)+,(R8)                  ;REMOVE LEFT PARENTHESIS FROM STACK
             13    11   01C7   409          BRB      180$                        ;
       53    20    D1   01C9   410 160$:    CMPL     #OPI_K_EOS,R3               ;END OF STATEMENT OPERATOR?
             69    13   01CC   411          BEQL     270$                        ;IF EQL YES
                        01CE   412
                        01CE   413 ;
                        01CE   414 ; STACK ITEM ONTO INTERMEDIATE PARSE STACK
                        01CE   415 ;
```

```
                01CE
78    52   D0   01CE   170$:   MOVL    R2,-(R8)                      ;STACK NEW ITEM
78    51   B0   01D1           MOVW    R1,-(R8)                      ;
78    53   B0   01D4           MOVW    R3,-(R8)                      ;
59    58   D1   01D7           CMPL    R8,R9                         ;PARSE STACK OVERFLOW?
      45   1B   01DA           BLEQU   240$                          ;IF LEQU YES
    FF12   31   01DC   180$:   BRW     10$                           ;
                01DF
                01DF   ;
                01DF   ; UNSTACK TRIAD FROM PARSE STACK
                01DF   ;
                01DF
      0A   BB   01DF   190$:   PUSHR   #^M<R1,R3>                    ;SAVE OPERATOR PARAMETERS
34    68   B1   01E1           CMPW    STK_W_TYPE(R8),#OPI_K_OPERAND ;TOP ITEM ON STACK OPERAND?
      44   19   01E4           BLSS    255$                          ;IF GTR NO
54    88   7D   01E6           MOVQ    (R8)+,R4                      ;REMOVE RIGHT HAND OPERAND FROM STACK
52    88   7D   01E9           MOVQ    (R8)+,R2                      ;REMOVE OPERATOR FROM STACK
52    02   B1   01EC           CMPW    #OPI_K_NOT,R2                 ;BOOLEAN NOT?
      14   13   01EF           BEQL    220$                          ;IF EQL YES
34    68   B1   01F1           CMPW    STK_W_TYPE(R8),#OPI_K_OPERAND ;TOP ITEM ON STACK OPERAND?
      0C   18   01F4           BGEQ    210$                          ;IF YES
52    0E   B1   01F6           CMPW    #OPI_K_NEG,R2                 ;OPERATOR UNARY MINUS?
      0A   13   01F9           BEQL    220$                          ;IF EQL YES
52    10   B1   01FB           CMPW    #OPI_K_POS,R2                 ;OPERATOR INTEGER NOOP?
      2A   12   01FE           BNEQ    255$                          ;IF NEQ NO
      03   11   0200           BRB     220$                          ;
50    88   7D   0202   210$:   MOVQ    (R8)+,R0                      ;REMOVE LEFT HAND OPERAND FROM STACK
78    59   D0   0205   220$:   MOVL    R9,-(R8)                      ;PUSH ADDRESS OF STACK OPERAND
78    36   9A   0208           MOVZBL  #OPI_K_STACK,-(R8)            ;PUSH OPERAND INDEX VALUE
58    59   D1   020B           CMPL    R9,R8                         ;TRIAD STACK OVERFLOW?
      0E   1A   020E           BGTRU   245$                          ;IF GTRU YES
89    54   7D   0210           MOVQ    R4,(R9)+                      ;PUSH RIGHT HAND OPERAND
89    52   3C   0213           MOVZWL  R2,(R9)+                      ;PUSH OPERATOR INDEX
89    50   7D   0216           MOVQ    R0,(R9)+                      ;PUSH LEFT HAND OPERAND
      0A   BA   0219   230$:   POPR    #^M<R1,R3>                    ;RETRIEVE OPERATOR PARAMETERS
    FF7B   31   021B           BRW     140$                          ;
                021E
                021E   ;
                021E   ; EXPRESION TOO COMPLEX
                021E   ;
                021E
5E    08   C0   021E   245$:   ADDL    #8,SP                         ;POP SAVED R1,R3 OFF STACK
                0221   240$:   STATUS  COMPLX                        ;SET COMPLEX EXPRESSION STATUS
      0A   11   0228           BRB     260$                          ;
                022A
                022A   ;
                022A   ; EXPRESSION SYNTAX ERROR
                022A   ;
                022A
5E    08   C0   022A   255$:   ADDL    #8,SP                         ;POP SAVED R1,R3 OFF STACK
                022D   250$:   STATUS  EXPSYN                        ;SET EXPRESSION SYNTAX ERROR STATUS
    01AF   31   0234   260$:   BRW     EXIT                          ;AND EXIT
                0237
                0237   ;
                0237   ; END OF STATEMENT - CHECK FOR VALID PARSE
                0237   ;
                0237
```

```
                    34    68   B1  0237    473 270$:    CMPW    STK_W_TYPE(R8),#OPI_K_OPERAND ;TOP ITEM ON STACK OPERAND?
                          F1   19  023A    474          BLSS    250$                      ;ERROR IF NOT
                    B9    88   7D  023C    475          MOVQ    (R8)+,(R9)+               ;MOVE RIGHT HAND OPERAND TO TRIAD STACK
                    68    1E   B1  023F    476          CMPW    #OPI_K_SOS,STK_W_TYPE(R8) ;TOP OF STACK START OF STATEMENT?
                          E9   12  0242    477          BNEQ    250$                      ;IF NEQ NO
                    69    26   9A  0244    478          MOVZBL  #OPI_K_STORE,(R9)         ;INSERT STORE OPERATOR
                        00B9   31  0247    479          BRW     EVALUATE                  ;EVALUATE THE EXPRESSION
```

```
                            024A         .SBTTL  SEARCH OPERATOR TABLE
                            024A   ;+
                            024A   ; FIND_OPERATOR - SEARCH OPERATOR TABLE
                            024A   ;
                            024A   ; THIS ROUTINE SEARCHES THE OPERATOR TABLE, GIVEN A STRING, TO
                            024A   ; LOCATE THE OPERATOR PRECEDENCE AND INDEX.
                            024A   ;
                            024A   ; INPUTS:
                            024A   ;
                            024A   ;       R1/R2 = DESCRIPTOR OF STRING
                            024A   ;
                            024A   ; OUTPUTS:
                            024A   ;
                            024A   ;       R0 = STATUS
                            024A   ;       R1 = OPERATOR PRECEDENCE
                            024A   ;       R3 = OPERATOR INDEX
                            024A   ;
                            024A   ;       R4,R5 DESTROYED.
                            024A   ;-
                            024A   FIND_OPERATOR:
   50   FDB2  CF       9E   024A         MOVAB   OPTAB,R0               ;GET ADDRESS OF OPERATOR TABLE
          55  50       D0   024F   40$:  MOVL    R0,R5                  ;RETRIEVE ADDRESS OF NEXT OPERATOR ENTRY
          50  85       9A   0252         MOVZBL  (R5)+,R0               ;GET OFFSET TO NEXT OPERATOR ENTRY
              1D       13   0255         BEQL    80$                    ;IF EQL END OF TABLE
          50  55       C0   0257         ADDL    R5,R0                  ;CALCULATE ADDRESS OF NEXT ENTRY
          85  51       91   025A         CMPB    R1,(R5)+               ;LENGTH OF NAMES MATCH?
              F0       12   025D         BNEQ    40$                    ;IF NEQ NO
          53  51       7D   025F         MOVQ    R1,R3                  ;COPY OPERATOR DESCRIPTOR
          85  84       91   0262   50$:  CMPB    (R4)+,(R5)+            ;CHARACTERS MATCH?
              E8       12   0265         BNEQ    40$                    ;IF NEQ NO
           F8 53       F5   0267         SOBGTR  R3,50$                 ;ANY MORE CHARACTERS TO MATCH?
          51  85       9A   026A         MOVZBL  (R5)+,R1               ;GET OPERATOR PRECEDENCE VALUE
          53  65       9A   026D         MOVZBL  (R5),R3                ;GET OPERATOR INDEX VALUE
          50  01       D0   0270         MOVL    #1,R0                  ;SUCCESS
                       05   0273         RSB
                            0274   80$:  STATUS  IVOPER                 ;REPORT INVALID OPERATOR
                       05   027B         RSB
```

```
                        027C            .SBTTL  PARSE_OPERAND - PARSE OPERAND TOKEN
                        027C    ;+
                        027C    ; PARSE_OPERAND - PARSE OPERAND TOKEN
                        027C    ;
                        027C    ; THIS ROUTINE IS CALLED TO PARSE AN OPERAND TOKEN AND RETURN
                        027C    ; THE OPERAND PARAMETERS DESCRIBING THE OPERAND.
                        027C    ;
                        027C    ; INPUTS:
                        027C    ;
                        027C    ;       R1/R2 = STRING DESCRIPTOR OF OPERAND
                        027C    ;
                        027C    ; OUTPUTS:
                        027C    ;
                        027C    ;       R0 = STATUS
                        027C    ;
                        027C    ;       THE CALLER'S STACK MAY BE UPDATED IN ORDER TO SAVE
                        027C    ;       INTERMEDIATE STRINGS OR BINARY VALUES.
                        027C    ;-
                        027C
                        027C    PARSE_OPERAND:
        7E     62    9A 027C            MOVZBL  (R2),-(SP)              ;SAVE STARTING CHARACTER OF SYMBOL
            FD7E'    30 027F            BSBW    DCL$COMPRESS            ;COMPRESS QUOTED STRING
        53     34    9A 0282            MOVZBL  #OPI_K_STRING,R3        ;ASSUME CHARACTER STRING LITERAL
        8E     22    D1 0285            CMPL    #^A/"/,(SP)+            ;STRING LITERAL?
               54    13 0288            BEQL    170$                    ;IF EQL YES
        62     25    91 028A            CMPB    #^A/%/,(R2)             ;NUMERIC RADIX OPERATOR?
               53    13 028D            BEQL    105$                    ;BR IF YES-ITS A NUMBER
        62     30    91 028F            CMPB    #^A/0/,(R2)             ;POSSIBLY NUMERIC CONSTANT?
               05    1A 0292            BGTRU   90$                     ;IF GTRU NO
        62     39    91 0294            CMPB    #^A/9/,(R2)             ;NUMERIC CONSTANT?
               49    1E 0297            BGEQU   105$                    ;IF GEQU YES
               57    DD 0299    90$:    PUSHL   R7                      ;SAVE ADDRESS OF STACK STORAGE
               0C    E1 029B            BBC     #WRK_V_COMMENT,-        ;SKIP IF NOT IN COMMENT
     03 F0 AA           029D                    WRK_B_FLAGS(R10),92$    ; MAKE F$VER EXPR EVAL WORK IN COMMENTS
            FD5D'    30 02A0            BSBW    DCL$UPCASE              ;UPCASE THE SYMBOL NAME
        56     51    7D 02A3    92$:    MOVQ    R1,R6                   ;COPY DESCRIPTOR FOR DCL$LEXIF
            FD57'    30 02A6            BSBW    DCL$SEARCH              ;SEARCH SYMBOL TABLE FOR VALUE
           03  50    E8 02A9            BLBS    R0,95$                  ;IF LBC SEARCH FAILURE
            FD51'    30 02AC            BSBW    DCL$LEXIF               ;GET VALUE OF LEXICAL FUNCTION
            57 8ED0     02AF    95$:    POPL    R7                      ;RESTORE ADDRESS OF STACK STORAGE
           46  50    E9 02B2            BLBC    R0,110$                 ;BRANCH IF NOT A FUNCTION
               52    D5 02B5            TSTL    R2                      ;BINARY OR STRING?
               17    13 02B7            BEQL    96$                     ;BRANCH IF BINARY
            50 8ED0     02B9            POPL    R0                      ;PICK UP RETURN ADDRESS
        5E     51    C2 02BC            SUBL    R1,SP                   ;ALLOCATE SPACE FOR STRING ON STACK
               50    DD 02BF            PUSHL   R0                      ;RESTORE RETURN ADDRESS
        56     51    D0 02C1            MOVL    R1,R6                   ;SAVE STRING SIZE
 04 AE  62     51    28 02C4            MOVC    R1,(R2),4(SP)           ;COPY STRING OPERAND TO STACK
        51     56    D0 02C9            MOVL    R6,R1                   ;RESTORE STRING SIZE
     52     04 AE    9E 02CC            MOVAB   4(SP),R2                ;SET ADDRESS OF COPIED STRING
            50 8ED0     02D0    96$:    POPL    R0                      ;PICK UP RETURN ADDRESS
        7E     51    7D 02D3            MOVQ    R1,-(SP)                ;PUSH VALUE DESCRIPTOR ONTO STACK
        52     5E    D0 02D6            MOVL    SP,R2                   ;SET ADDRESS OF ON-STACK OPERAND
               50    DD 02D9            PUSHL   R0                      ;RESTORE RETURN ADDRESS
        53     36    9A 02DB            MOVZBL  #OPI_K_STACK,R3         ;SET TYPE OF ITEM
        50     01    D0 02DE    170$:   MOVL    #1,R0                   ;SUCCESS
                     05 02E1            RSB
```

```
                          02E2     576 ;
                          02E2     577 ; PARSE NUMERIC OPERAND
                          02E2     578 ;
         52    51     7D  02E2     579 105$:     MOVQ     R1,R2                     ;COPY DESCRIPTOR
   51    00AE CB      9A  02E5     580           MOVZBL   PRC_B_DEFRADIX(R11),R1    ;SET RADIX FOR CONVERSION
              FD13'   30  02EA     581           BSBW     DCL$CNVASCBIN             ;CONVERT TO BINARY LONGWORD
                 04   12  02ED     582           BNEQ     108$                      ;BRANCH IF CONVERSION ERROR
                 52   D4  02EF     583           CLRL     R2                        ;MARK VALUE IS BINARY
                 DD   11  02F1     584           BRB      96$                       ;AND STORE IT ON THE STACK
                          02F3     585
                          02F3     586 108$:     STATUS   IVCHAR                    ;ERROR CONVERTING NUMBER
                      05  02FA     587           RSB
                          02FB     588 110$:     STATUS   UNDSYM                    ;SET UNDEFINED SYMBOL STATUS
                      05  0302     589           RSB
```

```
                        0303   591 ;
                        0303   592 ; EVALUATE THE EXPRESSION WHICH NOW CONSISTS OF TRIADS ORDERED
                        0303   593 ; BY PRECEDENCE, DESCRIBING THE ARGUMENTS AND OPERATORS IN THE
                        0303   594 ; ORDER THEY ARE TO BE EVALUATED.
                        0303   595 ;
                        0303   596
                        0303   597 EVALUATE:                                         ;EVALUATE EXPRESSION
  59   FE10 C7     9E   0303   598          MOVAB   -TRIADSTKSIZ-PARSESTKSIZ(R7),R9 ;GET STARTING ADDRESS OF TRIAD STACK
       5C   59     D0   0308   599 10$:     MOVL    R9,AP                     ;ADDRESS TO STORE RESULT
          00E9     30   030B   600          BSBW    FETCH                     ;FETCH RIGHT HAND OPERAND PARAMETERS
       56   89     D0   030E   601          MOVL    (R9)+,R6                  ;FETCH OPERATOR INDEX
       28   56     D1   0311   602          CMPL    R6,#OPI_K_EQS             ;STRING OPERATOR?
            5A     18   0314   603          BGEQ    20$                       ;IF GEQ YES
       26   56     D1   0316   604          CMPL    R6,#OPI_K_STORE           ;STORE OPERATOR?
            53     13   0319   605          BEQL    40$                       ;IF EQL YES
       06   56     D1   031B   606          CMPL    R6,#OPI_K_ADD             ;ADD?
            05     13   031E   607          BEQL    12$                       ;BRANCH IF SO
       08   56     D1   0320   608          CMPL    R6,#OPI_K_SUB             ;SUB?
            26     12   0323   609          BNEQ    15$                       ;BRANCH IF NOT
                        0325   610
                        0325   611 ;
                        0325   612 ; (+) OR (-) OPERATORS.  DECIDE IF STRING OR ARITHMETIC OPERATOR SHOULD BE
                        0325   613 ; APPLIED.  ASSUME ARITHMETIC UNLESS BOTH SIDES ARE STRINGS
                        0325   614 ;
                        0325   615
          00D9     30   0325   616 12$:     BSBW    OPERAND                   ;GET RIGHT HAND OPERAND DESCRIPTOR
            21     13   0328   617          BEQL    15$                       ;IF BINARY, DO AN INTEGER ADD
       54   51     7D   032A   618          MOVQ    R1,R4                     ;SAVE RIGHT HAND OPERAND STRING DESCRIPTOR
          00C7     30   032D   619          BSBW    FETCH                     ;FETCH LEFT HAND OPERAND PARAMETERS
          00CE     30   0330   620          BSBW    OPERAND                   ;GET LEFT HAND OPERAND DESCRIPTOR
            0D     13   0333   621          BEQL    14$                       ;IF BINARY, ATTEMPT ARITHMETIC OPERATION
       52   51     7D   0335   622          MOVQ    R1,R2                     ;SAVE LEFT HAND OPERAND DESCRIPTOR
     044D'CF46     16   0338   623          JSB     W^STRINGDISP-OPI_K_ADD[R6] ;PERFORM STRING OPERATION
                        033D   624                                            ;(RESULTANT STRING PLACED ONTO STACK)
       6C   54     7D   033D   625          MOVQ    R4,(AP)                   ;STORE RESULTANT STRING DESCRIPTOR
            C6     11   0340   626          BRB     10$                       ;
       59   14     C2   0342   627 14$:     SUBL    #8+4+8,R9                 ;BACKUP TRIAD STACK
          00AF     30   0345   628          BSBW    FETCH                     ;RE-FETCH LEFT HAND OPERAND
       56   89     D0   0348   629          MOVL    (R9)+,R6                  ;RE-FETCH OPERATOR INDEX
                        034B   630
                        034B   631 ;
                        034B   632 ; ARITHMETIC RELATIONAL OR BOOLEAN OPERATOR
                        034B   633 ;
                        034B   634
          00CE     30   034B   635 15$:     BSBW    NUMERIC                   ;CONVERT RIGHT HAND OPERAND TO NUMERIC
       54   50     D0   034E   636          MOVL    R0,R4                     ;SAVE RIGHT HAND OPERAND VALUE
          00A3     30   0351   637          BSBW    FETCH                     ;FETCH LEFT HAND OPERAND PARAMETERS
       02   56     D1   0354   638          CMPL    R6,#OPI_K_NOT             ;BOOLEAN NOT OPERATOR?
            44     13   0357   639          BEQL    30$                       ;IF EQL YES
       0E   56     D1   0359   640          CMPL    R6,#OPI_K_NEG             ;INTEGER NEGATION?
            3F     13   035C   641          BEQL    30$                       ;IF EQL YES
       10   56     D1   035E   642          CMPL    R6,#OPI_K_POS             ;INTEGER NOOP?
            3A     13   0361   643          BEQL    30$                       ;IF EQL YES
          00B6     30   0363   644          BSBW    NUMERIC                   ;CONVERT LEFT HAND OPERAND TO NUMERIC
       52   50     D0   0366   645          MOVL    R0,R2                     ;SAVE LEFT HAND OPERAND VALUE
       54   52     D1   0369   646          CMPL    R2,R4                     ;COMPARE RIGHT AND LEFT HAND OPERANDS
            2F     11   036C   647          BRB     30$                       ;
```

```
                            036E   648
                  3B   11   036E   649 40$:     BRB     41$                         ;CONTINUE TO STORE RESULT CODE
                            0370   650 ;
                            0370   651 ; STRING RELATIONAL OPERATOR
                            0370   652 ;
                            0370   653
                009C   30   0370   654 20$:     BSBW    STRING                      ;CONVERT RIGHT HAND OPERAND TO STRING
             F486 CA   DD   0373   655          PUSHL   WRK_L_EXPANDPTR(R10)        ;ASSUME RESULT IS IN EXPANSION BUFFER
        F486 CA   52   D1   0377   656          CMPL    R2,@RK_L_EXPANDPTR(R10)     ;IS RESULT IN EXPANSION BUFFER
                  05   12   037C   657          BNEQ    25$                         ;IF NOT, THEN SKIP
        F486 CA   51   C0   037E   658          ADDL    R1,WRK_L_EXPANDPTR(R10)     ;MOVE EXPANSION POINTER PAST STRING
             54   51   7D   0383   659 25$:     MOVQ    R1,R4                       ;SAVE RIGHT HAND OPERAND STRING DESCRIPTOR
                006E   30   0386   660          BSBW    FETCH                       ;FETCH LEFT HAND OPERAND PARAMETERS
                0083   30   0389   661          BSBW    STRING                      ;CONVERT LEFT HAND OPERAND TO STRING
             52   51   7D   038C   662          MOVQ    R1,R2                       ;SAVE LEFT HAND OPERAND DESCRIPTOR
             56   16   C2   038F   663          SUBL    #OPI_K_EQS-OPI_K_EQ,R6      ;NORMALIZE OPERATOR INDEX
             F486 CA 8ED0   0392   664          POPL    WRK_L_EXPANDPTR(R10)        ;RESTORE EXPANSION BUFFER POINTER
65   54   00   63   52   2D   0397   665          CMPC5   R2,(R3),#0,R4,(R5)          ;COMPARE RIGHT AND LEFT HAND OPERANDS
        04C2'CF46   16   039D   666 30$:     JSB     W^DISPATCH[R6]              ;EXECUTE OPERATOR SPECIFIC ROUTINE
             6C   54   D0   03A2   667          MOVL    R4,(AP)                     ;STORE OPERATION RESULT
               04 AC   D4   03A5   668          CLRL    4(AP)                       ;INDICATE RESULT IS BINARY
                FF5D   31   03A8   669          BRW     10$                         ;
                            03AB   670
                            03AB   671 ;
                            03AB   672 ; STORE RESULT OPERATOR
                            03AB   673 ;
                  54   10   03AB   674 41$:     BSBB    OPERAND                     ;GET DESCRIPTOR OF RIGHT-HAND OPERAND
                  30   13   03AD   675          BEQL    50$                         ;BRANCH IF BINARY RESULT
        50  F492 CA    9E   03AF   676          MOVAB   WRK_G_BUFFER(R10),R0        ;GET ADDRESS OF EXPANSION BUFFER
   50   00000400 8F    C0   03B4   677          ADDL    #WRK_C_CMDBUFSIZ,R0         ;FIND END OF BUFFER
        50  F486 CA    C2   03BB   678          SUBL    WRK_L_EXPANDPTR(R10),R0     ;CALCULATE HOW MUCH IS LEFT
             51   50   D1   03C0   679          CMPL    R0,R1                       ;WILL RESULT FIT?
                  09   14   03C3   680          BGTR    45$                         ;YES, THEN STORE IT AWAY
                            03C5   681          STATUS  BUFOVF                      ;SET OVERFLOW STATUS
                  18   11   03CC   682          BRB     EXIT                        ;EXIT WITH ERROR
             56   51   D0   03CE   683 45$:     MOVL    R1,R6                       ;SAVE LENGTH OF STRING
   F486 DA   62   51   28   03D1   684          MOVC    R1,(R2),@WRK_L_EXPANDPTR(R10) ;COPY IT TO THE EXPANSION BUFFER
             51   56   D0   03D7   685          MOVL    R6,R1                       ;RETURN LENGTH OF RESULT STRING
        52  F486 CA    D0   03DA   686          MOVL    WRK_L_EXPANDPTR(R10),R2     ;AND ADDRESS OF RESULT STRING
                            03DF   687 50$:     STATUS  NORMAL                      ;SET NORMAL COMPLETION STATUS
          5E   0C A7   9E   03E6   688 EXIT:    MOVAB   LOCALSIZ(R7),SP             ;DEALLOCATE SCRATCH STACKS
             13F8 8F   BA   03EA   689          POPR    #^M<R3,R4,R5,R6,R7,R8,R9,AP> ;RESTORE REGISTERS
               05 50   E9   03EE   690          BLBC    R0,90$                      ;BRANCH IF ERROR EXIT
        F48A CA   53   D0   03F1   691          MOVL    R3,WRK_L_MARKPTR(R10)       ;LEAVE MARKPTR SET TO START OF EXPR.
                       05   03F6   692 90$:     RSB                                 ;
```

```
                                    03F7     694          .SBTTL  FETCH NEXT OPERAND FROM TRIAD STACK
                                    03F7     695  ;
                                    03F7     696  ; FETCH - FETCH NEXT OPERAND FROM TRIAD STACK
                                    03F7     697  ;
                                    03F7     698
                50     89     3C    03F7     699  FETCH:  MOVZWL  (R9)+,R0                    ;GET TYPE OF OPERAND
                51     89     3C    03FA     700          MOVZWL  (R9)+,R1                    ;GET LENGTH OF OPERAND
                52     89     D0    03FD     701          MOVL    (R9)+,R2                    ;GET ADDRESS OF OPERAND
                              05    0400     702          RSB                                 ;
```

```
                        0401               .SBTTL  GET OPERAND DESCRIPTOR
                        0401       ;
                        0401       ; OPERAND - GET OPERAND DESCRIPTOR FROM OPERAND PARAMETERS
                        0401       ;
                        0401       ; Z BIT SET ON R2 (ADDRESS), Z=1 INDICATES BINARY, Z=0 INDICATES STRING
                        0401       ;
                        0401       OPERAND:
       50    36    D1   0401               CMPL    #OPI_K_STACK,R0         ;STACK OPERAND?
             06    12   0404               BNEQ    10$                     ;BRANCH IF NOT
       51    62    7D   0406               MOVQ    (R2),R1                 ;GET OPERAND DESCRIPTOR
       50    34    9A   0409               MOVZBL  #OPI_K_STRING,R0        ;MARK OPERAND NOW A STRING
             52    D5   040C       10$:    TSTL    R2                      ;SET CONDITION CODES ON TYPE
                   05   040E               RSB
```

```
              040F          .SBTTL  CONVERT STRING PARAMETERS TO STRING OPERAND
              040F   ;
              040F   ; STRING - CONVERT OPERAND TO CHARACTER STRING
              040F   ;
              040F   ; IF THE OPERAND IS EVALUATED, THEN RETURN ITS DESCRIPTOR IF ITS
              040F   ; A STRING, AND IF NUMERIC, CONVERT IT TO ASCII DECIMAL.
              040F   ; IF THE OPERAND IS AN UNEVALUATED CHARACTER OR NUMERIC STRING LITERAL,
              040F   ; SIMPLY RETURN WITH THE DESCRIPTOR.
              040F   ;
              040F
              040F  STRING:                                        ;STRING OPERAND
   F0    10   040F          BSBB    OPERAND                        ;GET OPERAND DESCRIPTOR
   08    12   0411          BNEQ    10$                            ;IF STRING, EXIT WITH DESCRIPTOR
   53    DD   0413          PUSHL   R3                             ;SAVE R3
FBE8'    30   0415          BSBW    DCL$CVT_STRING                 ;CONVERT TO STRING
   53  8ED0   0418          POPL    R3                             ;RESTORE R3
         05   041B  10$:    RSB                                    ;
```

```
                    041C          .SBTTL  CONVERT OPERAND PARAMETERS TO NUMERIC VALUE
                    041C    ;
                    041C    ; NUMERIC - CONVERT OPERAND PARAMETERS TO NUMERIC VALUE
                    041C    ;
                    041C
                    041C    NUMERIC:                                  ;NUMERIC OPERAND
                    041C    ;
                    041C    ; IF THE OPERAND IS ALREADY BINARY FROM A PREVIOUS EVALUATION, RETURN IT
                    041C    ;
          E3    10  041C          BSBB    OPERAND                     ;GET OPERAND DESCRIPTOR
          1A    13  041E          BEQL    30$                         ;IF BINARY, RETURN WITH VALUE IN R1
                    0420    ;
                    0420    ; CONVERT A NUMERIC STRING TO A BINARY LONGWORD
                    0420    ;
7E    00AE CB   9A  0420          MOVZBL  PRC_B_DEFRADIX(R11),-(SP) ;SAVE DEFAULT RADIX
        50   34 D1  0425          CMPL    #OPI_K_STRING,R0            ;IS THIS A CHARACTER STRING LITERAL?
             05 13  0428          BEQL    10$                         ;IF SO, USE DEFAULT RADIX
  00AE CB    01 9A  042A          MOVZBL  #PRC_K_DEC,PRC_B_DEFRADIX(R11) ;OTHERWISE, USE DECIMAL RADIX
           FC65 30  042F    10$:  BSBW    DCL$CVT_BINARY              ;CONVERT STRING TO BINARY
  00AE CB    8E F6  0432          CVTLB   (SP)+,PRC_B_DEFRADIX(R11) ;RESTORE DEFAULT RADIX
          04 50 E9  0437          BLBC    R0,85$                      ;BRANCH IF ERROR DETECTED
        50   51 D0  043A    30$:  MOVL    R1,R0                       ;SET VALUE
                05  043D          RSB                                 ;
                    043E
          FBBF' 30  043E    85$:  BSBW    DCL$MARK                    ;MARK START OF ERROR SEGMENT
        50   62 90  0441          MOVB    (R2),R0                     ;PICK UP FIRST BYTE OF STRING
          FBB9' 30  0444          BSBW    DCL$PUTCHAR                 ;WRITE CHARACTER INTO BUFFER
                    0447                                              ;SO THAT ERROR REPORTING CAN
                    0447                                              ;DISPLAY THE CHARACTER
             8E D5  0447          TSTL    (SP)+                       ;CLEAN STACK OF PREVIOUS CALLER
                    0449          STATUS  IVCHAR                      ;INVALID CHARACTER
           FF93 31  0450          BRW     EXIT                        ;RETURN TO CALLER'S CALLER
```

```
                   0453   769          .SBTTL  DISPATCH STRING OPERATION FUNCTION
                   0453   770 ;
                   0453   771 ; DISPATCH FUNCTIONS WHICH RESULT IN STRINGS
                   0453   772 ;
                   0453   773 STRINGDISP:
          02   11  0453   774          BRB     CONCAT                  ;STRING CONCATENATION
          36   11  0455   775          BRB     REDUCE                  ;STRING REDUCTION
```

```
                                  0457                .SBTTL  STRING CONCATENATION OPERATOR
                                  0457        ;
                                  0457        ; THIS ROUTINE PROCESSES THE STRING CONCATENATION OPERATOR AND STORES
                                  0457        ; THE RESULTANT STRING ON THE CALLER'S STACK.
                                  0457        ;
                                  0457        ; INPUTS:
                                  0457        ;
                                  0457        ;       R2/R3 = STRING DESCRIPTOR OF LEFT-HAND SIDE
                                  0457        ;       R4/R5 = STRING DESCRIPTOR OF RIGHT-HAND SIDE
                                  0457        ;
                                  0457        ; OUTPUTS:
                                  0457        ;
                                  0457        ;       R4/R5 = STRING DESCRIPTOR OF RESULTANT STRING
                                  0457        ;
                                  0457        ;       R0-R3,R6 DESTROYED.
                                  0457        ;
                        56 8ED0   0457        CONCAT: POPL    R6                      ;POP CALLER'S RETURN ADDRESS
         50    54       52   C1   045A                ADDL3   R2,R4,R0                ;COMPUTE SIZE OF RESULTANT STRING
               5E       50   C2   045E                SUBL    R0,SP                   ;ALLOCATE SPACE ON CALLER'S STACK
  80 AE   0080 8F       00   0D   0461                PROBEW  #0,#128,-128(SP)        ;CHECK FOR STACK OVERFLOW
                        0A   12   0468                BNEQ    10$                     ;CONTINUE IF SUCCESSFUL
                                  046A                STATUS  BUFOVF                  ;SET STATUS
                      FF72   31   0471                BRW     EXIT                    ;EXIT WITH STATUS
                        50   DD   0474        10$:    PUSHL   R0                      ;SAVE SIZE OF RESULTANT STRING
               7E       54   7D   0476                MOVQ    R4,-(SP)                ;SAVE DESCRIPTOR OF RIGHT-SIDE
      0C AE    63       52   28   0479                MOVC    R2,(R3),12(SP)          ;COPY FIRST STRING ONTO STACK
               54       8E   7D   047E                MOVQ    (SP)+,R4                ;RESTORE DESCRIPTOR OF RIGHT-SIDE
         63    65       54   28   0481                MOVC    R4,(R5),(R3)            ;APPEND SECOND STRING TO FIRST
                        54 8ED0   0485                POPL    R4                      ;RESTORE SIZE OF RESULTANT STRING
               55       5E   D0   0488                MOVL    SP,R5                   ;SET ADDRESS OF RESULTANT STRING
                        66   17   048B                JMP     (R6)                    ;RETURN
```

```
                             048D                .SBTTL  STRING REDUCTION OPERATOR
                             048D        ;
                             048D        ; THIS ROUTINE PROCESSES THE STRING REDUCTION OPERATOR (-) AND STORES
                             048D        ; THE RESULTANT STRING ON THE CALLER'S STACK.
                             048D        ;
                             048D        ; INPUTS:
                             048D        ;
                             048D        ;       R2/R3 = STRING DESCRIPTOR OF LEFT-HAND SIDE
                             048D        ;       R4/R5 = STRING DESCRIPTOR OF RIGHT-HAND SIDE
                             048D        ;
                             048D        ; OUTPUTS:
                             048D        ;
                             048D        ;       R4/R5 = STRING DESCRIPTOR OF RESULTANT STRING
                             048D        ;
                             048D        ;       R0-R3,R6 DESTROYED.
                             048D        ;
                   0C     BB 048D        REDUCE: PUSHR   #^M<R2,R3>              ;SAVE LEFT SIDE OPERAND
 63    52    65    54     39 048F                MATCHC  R4,(R5),R2,(R3)         ;LOCATE SUBSTRING WITHIN STRING
                   28     12 0494                BNEQ    50$                     ;BRANCH IF NOT FOUND IN LEFT SIDE
             50    52     7D 0496                MOVQ    R2,R0                   ;SAVE DESCRIPTOR OF PIECE AFTER MATCH
             004C  8F     BA 0499                POPR    #^M<R2,R3,R6>           ;GET LEFT SIDE AND RETURN ADDRESS
       54    52    54     C3 049D                SUBL3   R4,R2,R4                ;COMPUTE SIZE OF RESULTANT STRING
       52    54    50     C3 04A1                SUBL3   R0,R4,R2                ;COMPUTE SIZE OF PIECE BEFORE MATCH
             5E    54     C2 04A5                SUBL    R4,SP                   ;ALLOCATE SPACE ON CALLER'S STACK
                   13     BB 04A8                PUSHR   #^M<R0,R1,R4>           ;SAVE REGISTERS
    0C AE    63    52     28 04AA                MOVC    R2,(R3),12(SP)          ;COPY PIECE BEFORE MATCH INTO RESULT
             50    8E     7D 04AF                MOVQ    (SP)+,R0                ;GET DESCRIPTOR OF PIECE AFTER MATCH
       63    61    50     28 04B2                MOVC    R0,(R1),(R3)            ;APPEND PIECE AFTER MATCH INTO RESULT
                   54   8ED0 04B6                POPL    R4                      ;RESTORE SIZE OF RESULTANT STRING
             55    5E     D0 04B9                MOVL    SP,R5                   ;SET ADDRESS OF RESULTANT STRING
                   66     17 04BC                JMP     (R6)                    ;RETURN
                             04BE        ;
                             04BE        ; SUBSTRING NOT FOUND IN STRING - RETURN MAIN STRING INTACT
                             04BE        ;
             54    8E     7D 04BE        50$:    MOVQ    (SP)+,R4                ;RETURN DESCRIPTOR OF LEFT SIDE
                          05 04C1                RSB
```

```
                        04C2    846             .SBTTL  DISPATCH BINARY/LOGICAL OPERATOR
                        04C2    847     ;
                        04C2    848     ; DISPATCH BINARY/LOGICAL OPERATOR FUNCTION
                        04C2    849     ;
                        04C2    850     ; INPUTS:
                        04C2    851     ;
                        04C2    852     ;       R2/R3 = LEFT HAND OPERAND
                        04C2    853     ;       R4/R5 = RIGHT HAND OPERAND
                        04C2    854     ;
                        04C2    855     ; OUTPUTS:
                        04C2    856     ;
                        04C2    857     ;       R4/R5 = RESULT VALUE
                        04C2    858     ;
                        04C2    859
                        04C2    860     DISPATCH:                                       ;
              1E   11   04C2    861             BRB     AND                             ;BOOLEAN AND
              23   11   04C4    862             BRB     NOT                             ;BOOLEAN NOT
              25   11   04C6    863             BRB     OR                              ;BOOLEAN OR
              27   11   04C8    864             BRB     ADD                             ;INTEGER ADD
              29   11   04CA    865             BRB     SUB                             ;INTEGER SUBTRACT
              2C   11   04CC    866             BRB     MUL                             ;INTEGER MULTIPLY
              2E   11   04CE    867             BRB     DIV                             ;INTEGER DIVIDE
              41   11   04D0    868             BRB     NEG                             ;INTEGER NEGATION
              43   11   04D2    869             BRB     POS                             ;INTEGER NOOP
              42   11   04D4    870             BRB     EQL                             ;EQUAL
              44   11   04D6    871             BRB     GEQ                             ;GREATER OR EQUAL
              46   11   04D8    872             BRB     GTR                             ;GREATER
              48   11   04DA    873             BRB     LEQ                             ;LESS OR EQUAL
              4A   11   04DC    874             BRB     LSS                             ;LESS
                        04DE    875     ;       BRB     NEQ                             ;NOT EQUAL
                        04DE    876
                        04DE    877     ;
                        04DE    878     ; NOT EQUAL
                        04DE    879     ;
                        04DE    880
              4D   12   04DE    881     NEQ:    BNEQ    SETRUE                          ;IF NEQ SETRUE RESULT
              48   11   04E0    882             BRB     SETFALSE
                        04E2    883
                        04E2    884     ;
                        04E2    885     ; BOOLEAN AND
                        04E2    886     ;
                        04E2    887
        52    52   D2   04E2    888     AND:    MCOML   R2,R2                           ;COMPLEMENT LEFT HAND OPERAND
        54    52   CA   04E5    889             BICL    R2,R4                           ;FORM BOOLEAN AND FUNCTION
                   05   04E8    890             RSB                                     ;
                        04E9    891
                        04E9    892     ;
                        04E9    893     ; BOOLEAN NOT
                        04E9    894     ;
                        04E9    895
        54    54   D2   04E9    896     NOT:    MCOML   R4,R4                           ;FORM BOOLEAN NOT FUNCTION
                   05   04EC    897             RSB                                     ;
                        04ED    898
                        04ED    899     ;
                        04ED    900     ; BOOLEAN OR
                        04ED    901     ;
                        04ED    902
```

```
          54   52   C8  04ED   903 OR:     BISL    R2,R4                 ;FORM BOOLEAN OR FUNCTION
                    05  04F0   904         RSB                           ;
                        04F1   905
                        04F1   906 ;
                        04F1   907 ; INTEGER ADD
                        04F1   908 ;
                        04F1   909
          54   52   C0  04F1   910 ADD:    ADDL    R2,R4                 ;FORM ARITHMETIC SUM
                    05  04F4   911         RSB                           ;
                        04F5   912
                        04F5   913 ;
                        04F5   914 ; INTEGER SUBTRACT
                        04F5   915 ;
                        04F5   916
     54   52   54   C3  04F5   917 SUB:    SUBL3   R4,R2,R4              ;FORM ARITHMETIC DIFFERENCE
                    05  04F9   918         RSB                           ;
                        04FA   919
                        04FA   920 ;
                        04FA   921 ; INTEGER MULTIPLY
                        04FA   922 ;
                        04FA   923
          54   52   C4  04FA   924 MUL:    MULL    R2,R4                 ;FORM ARITHMETIC PRODUCT
                    05  04FD   925         RSB                           ;
                        04FE   926
                        04FE   927 ;
                        04FE   928 ; INTEGER DIVIDE
                        04FE   929 ;
                        04FE   930
               54   D5  04FE   931 DIV:    TSTL    R4                    ;DIVIDE BY ZERO ATTEMPT?
               0C   12  0500   932         BNEQ    20$                   ;BR IF NO
                        0502   933         SETBIT  #31,R4                ;MAKE RESULT THE HIGHEST NEGATIVE NUMBER
               52   D5  0506   934         TSTL    R2                    ;SOURCE NEGATIVE?
               03   19  0508   935         BLSS    10$                   ;BR IF YES
          54   54   D2  050A   936         MCOML   R4,R4                 ;MAKE THE LARGEST POSITIVE NUMBER
                    05  050D   937 10$:    RSB                           ;
     54   52   54   C7  050E   938 20$:    DIVL3   R4,R2,R4              ;FORM ARITHMETIC QUOTIENT
                    05  0512   939         RSB                           ;
                        0513   940
                        0513   941
                        0513   942 ;
                        0513   943 ; INTEGER NEGATION
                        0513   944 ;
                        0513   945
          54   54   CE  0513   946 NEG:    MNEGL   R4,R4                 ;NEGATE OPERAND
                    05  0516   947         RSB                           ;
                        0517   948
                        0517   949 ;
                        0517   950 ; INTEGER NOOP
                        0517   951 ;
                        0517   952
                    05  0517   953 POS:    RSB                           ;NOOP
                        0518   954
                        0518   955 ;
                        0518   956 ; EQUAL
                        0518   957 ;
                        0518   958
               13   13  0518   959 EQL:    BEQL    SETRUE                ;IF EQL SETRUE RESULT
```

```
         0E  11  051A          BRB     SETFALSE
                 051C
                 051C  ;
                 051C  ; GREATER OR EQUAL
                 051C  ;
                 051C
         0F  18  051C  GEQ:    BGEQ    SETRUE                  ;IF GEQ SETRUE RESULT
         0A  11  051E          BRB     SETFALSE
                 0520
                 0520  ;
                 0520  ; GREATER
                 0520  ;
                 0520
         08  14  0520  GTR:    BGTR    SETRUE                  ;IF GTR SETRUE RESULT
         06  11  0522          BRB     SETFALSE
                 0524
                 0524  ;
                 0524  ; LESS OF EQUAL
                 0524  ;
                 0524
         07  15  0524  LEQ:    BLEQ    SETRUE                  ;IF LEQ SETRUE RESULT
         02  11  0526          BRB     SETFALSE
                 0528
                 0528  ;
                 0528  ; LESS
                 0528  ;
                 0528
         03  19  0528  LSS:    BLSS    SETRUE                  ;IF LSS SETRUE RESULT
                 052A
                 052A  ;
                 052A  ; RETURN FALSE RESULT
                 052A  ;
                 052A
                 052A  SETFALSE:
         54  D4  052A          CLRL    R4                      ;SET RESULT FALSE
             05  052C          RSB
                 052D
                 052D  ;
                 052D  ; RETURN TRUE RESULT
                 052D  ;
                 052D
     54  01  D0  052D  SETRUE: MOVL    #1,R4                   ;SET RESULT TRUE
             05  0530          RSB                             ;
                 0531
                 0531          .END
```

```
ADD                                000004F1 R       02      OPI_K_GT                          = 00000016
AND                                000004E2 R       02      OPI_K_GTS                         = 0000002C
CLI$_BUFOVF                      = 00038018                 OPI_K_LE                          = 00000018
CLI$_COMPLX                      = 00038020                 OPI_K_LES                         = 0000002E
CLI$_EXPSYN                      = 00038038                 OPI_K_LPAREN                      = 00000022
CLI$_IVCHAR                      = 00038050                 OPI_K_LT                          = 0000001A
CLI$_IVOPER                      = 00038068                 OPI_K_LTS                         = 00000030
CLI$_NORMAL                      = 00030001                 OPI_K_MUL                         = 0000000A
CLI$_UNDSYM                      = 00038140                 OPI_K_NE                          = 0000001C
CONCAT                             00000457 R       02      OPI_K_NEG                         = 0000000E
CURMODE                            00000002                 OPI_K_NES                         = 00000032
DCL$BACKUPMOVE                     ********    X    02      OPI_K_NOT                         = 00000002
DCL$BINEXPR                        00000091 RG      02      OPI_K_OPERAND                     = 00000034
DCL$CNVASCBIN                      ********    X    02      OPI_K_OR                          = 00000004
DCL$COMPRESS                       ********    X    02      OPI_K_POS                         = 00000010
DCL$CVT_BINARY                     00000097 RG      02      OPI_K_RPAREN                      = 00000024
DCL$CVT_STRING                     ********    X    02      OPI_K_SOS                         = 0000001E
DCL$EXPRADIX                       000000C3 RG      02      OPI_K_STACK                       = 00000036
DCL$EXPRESS                        000000C0 RG      02      OPI_K_STORE                       = 00000026
DCL$GETOKEN                        ********    X    02      OPI_K_STRING                      = 00000034
DCL$LEXIF                          ********    X    02      OPI_K_SUB                         = 00000008
DCL$MARK                           ********    X    02      OPP_K_ADD                         = 00000008
DCL$MOVBTOKN                       ********    X    02      OPP_K_AND                         = 00000005
DCL$MOVCHAR                        ********    X    02      OPP_K_DIV                         = 00000009
DCL$PUTCHAR                        ********    X    02      OPP_K_EOS                         = 00000001
DCL$SEARCH                         ********    X    02      OPP_K_EQ                          = 00000007
DCL$SETCHAR                        ********    X    02      OPP_K_EQS                         = 00000007
DCL$UPCASE                         ********    X    02      OPP_K_GE                          = 00000007
DISPATCH                           000004C2 R       02      OPP_K_GES                         = 00000007
DIV                                000004FE R       02      OPP_K_GT                          = 00000007
EQL                                00000518 R       02      OPP_K_GTS                         = 00000007
EVALUATE                           00000303 R       02      OPP_K_LE                          = 00000007
EXIT                               000003E6 R       02      OPP_K_LES                         = 00000007
FALSE                              00000085 R       02      OPP_K_LPAREN                      = 00000002
FETCH                              000003F7 R       02      OPP_K_LT                          = 00000007
FIND_OPERATOR                      0000024A R       02      OPP_K_LTS                         = 00000007
GEQ                                0000051C R       02      OPP_K_MUL                         = 00000009
GTR                                00000520 R       02      OPP_K_NE                          = 00000007
LEQ                                00000524 R       02      OPP_K_NEG                         = 0000000A
LIB$SCOPY_DXDX                     0000008E RG      02      OPP_K_NES                         = 00000007
LOCALSIZ                           0000000C                 OPP_K_NOT                         = 00000006
LSS                                00000528 R       02      OPP_K_OR                          = 00000004
MUL                                000004FA R       02      OPP_K_POS                         = 0000000A
NEG                                00000513 R       02      OPP_K_RPAREN                      = 00000003
NEQ                                000004DE R       02      OPP_K_SOS                         = 00000000
NESTLVL                            00000000                 OPP_K_STORE                       = 00000000
NOT                                000004E9 R       02      OPP_K_SUB                         = 00000008
NUMERIC                            0000041C R       02      OPTAB                               00000000 R       02
OPERAND                            00000401 R       02      OR                                  000004ED R       02
OPI_K_ADD                        = 00000006                 PARSESTKSIZ                       = 00000084
OPI_K_AND                        = 00000000                 PARSE_OPERAND                       0000027C R       02
OPI_K_DIV                        = 0000000C                 POS                                 00000517 R       02
OPI_K_EOS                        = 00000020                 PRC_B_CONTINUE                      000000F3
OPI_K_EQ                         = 00000012                 PRC_B_DEFRADIX                      000000AE
OPI_K_EQS                        = 00000028                 PRC_B_EXMDEPMOD                     000000AD
OPI_K_GE                         = 00000014                 PRC_B_EXMDEPWID                     000000AC
OPI_K_GES                        = 0000002A                 PRC_B_EXONLYL                       0000012D
```

```
PRC_B_FLAGS2                     000000AF          PRC_W_ASTRETN                    000000C8
PRC_B_IMGFLAG                    00000078          PRC_W_ASTSTATUS                  000000C4
PRC_B_OUTFLAGS                   0000012C          PRC_W_ATTMBX                     0000007A
PRC_B_PROMPTLEN                  000000F0          PRC_W_FLAGS                      00000068
PRC_C_LENGTH                     00000534          PRC_W_INPCHAN                    00000064
PRC_G_COMMANDS                   00000133          PRC_W_ONLEVEL                    0000006A
PRC_G_PROMPT                     000000F4          PRC_W_OUTIFI                     00000114
PRC_K_DEC                      = 00000001          PRC_W_OUTISI                     00000116
PRC_K_LENGTH                     00000534          PRC_W_OUTMBXCHN                  000000CA
PRC_L_CURRKEY                    00000048          PRC_W_OUTMBXREF                  000000CE
PRC_L_EXMDEPADR                  000000A8          PRC_W_OUTMBXSIZ                  000000CC
PRC_L_EXTARG                     00000094          PRC_W_PMPTCTRL                   000000F1
PRC_L_EXTBLK                     0000008C          PRC_W_WAITIOSB                   00000066
PRC_L_EXTCOD                     0000009C          REDUCE                           0000048D R     02
PRC_L_EXTHND                     00000090          REQMODE                          00000001
PRC_L_EXTPRM                     00000098          RESULT                           00000004
PRC_L_IDFLNK                     000000BC          SETFALSE                         0000052A R     02
PRC_L_IMGACTSTS                  00000080          SETRUE                           0000052D R     02
PRC_L_INDCLOCK                   0000007C          STK_K_LENGTH                     00000008
PRC_L_INDEPTH                    0000005C          STK_L_ADDR                       00000004
PRC_L_INDFAB                     0000001C          STK_W_PREC                       00000002
PRC_L_INDINPRAB                  00000014          STK_W_SIZE                       00000002
PRC_L_INDOUTRAB                  00000018          STK_W_TYPE                       00000000
PRC_L_INPRAB                     00000008          STRING                           0000040F R     02
PRC_L_LASTKEY                    0000004C          STRINGDISP                       00000453 R     02
PRC_L_LSTSTATUS                  000000B0          SUB                              000004F5 R     02
PRC_L_ONCTLY                     000000B8          TRIADSTKSIZ                    = 0000016C
PRC_L_ONERROR                    0000006C          TRUE                             00000081 R     02
PRC_L_OUTOFBAND                  000000B4          TRUSYM                           0000008A R     02
PRC_L_OUTRAB                     0000000C          WRK_B_CMDOPT                     FFFFFFC3
PRC_L_OUTRABCTX                  00000118          WRK_B_MAXPARM                    FFFFFFD0
PRC_L_PPFLIST                    00000070          WRK_B_MINPARM                    FFFFFFD1
PRC_L_RECALLPTR                  0000012F          WRK_B_PARMCNT                    FFFFFFCE
PRC_L_RESTART                    00000058          WRK_B_PARMSUM                    FFFFFFCF
PRC_L_SAVAP                      00000000          WRK_B_RECALLCNT                  FFFFFFC5
PRC_L_SAVFP                      00000004          WRK_B_VALLEV                     FFFFFFC4
PRC_L_SEVERITY                   00000050          WRK_B_VERBTYP                    FFFFFFC2
PRC_L_SPWN                       000000C0          WRK_C_CMDBUFSIZ                = 00000400
PRC_L_STACKLM                    000000A4          WRK_C_LENGTH                     FFFFF486
PRC_L_STACKPT                    000000A0          WRK_G_BUFFER                     FFFFF492
PRC_L_STATUS                     00000054          WRK_G_INPBUF                     FFFFF896
PRC_L_STS                        00000084          WRK_G_RESULT                     FFFFF9B6
PRC_L_STV                        00000088          WRK_K_LENGTH                     FFFFF486
PRC_L_SYMBOL                     00000060          WRK_L_CHARPTR                    FFFFF48E
PRC_L_TMBX                       00000074          WRK_L_DISALLOW                   FFFFFFE6
PRC_L_TRMLIST                    00000010          WRK_L_ERRORRTN                   FFFFF9AE
PRC_Q_ALLOCREG                   00000020          WRK_L_EXPANDPTR                  FFFFF486
PRC_Q_COMMAND                    000000E0          WRK_L_IMAGE                      FFFFFFE2
PRC_Q_FLUSHTIME                  000000D0          WRK_L_MARKPTR                    FFFFF48A
PRC_Q_GLOBAL                     00000028          WRK_L_PAROUT                     FFFFFFD2
PRC_Q_IMAGENAME                  000000D8          WRK_L_PMPTADDR                   FFFFF9A2
PRC_Q_KEYPAD                     00000040          WRK_L_PROMPTRTN                  FFFFF9A6
PRC_Q_LABEL                      00000030          WRK_L_PROPTR                     FFFFFFC6
PRC_Q_LOCAL                      00000038          WRK_L_QUABLK                     FFFFFFCA
PRC_Q_SAVEPRIV                   000000E8          WRK_L_READRTN                    FFFFF9AA
PRC_T_OUTDVI                     0000011C          WRK_L_RECALLPTR                  FFFFFFEA
PRC_W_ASTIOSB                    000000C6          WRK_L_RSLEND                     FFFFFFB6
```

```
WRK_L_RSLNXT                       FFFFFFBA
WRK_L_SAVAP                        FFFFFFF8
WRK_L_SAVFP                        FFFFFFFC
WRK_L_SAVSP                        FFFFFFF4
WRK_L_SIGNALRTN                    FFFFFFD6
WRK_L_SPECRTN                      FFFFF9B2
WRK_L_TAB_VEC                      FFFFFFDE
WRK_L_VERB                         FFFFFFBE
WRK_M_STAR                       = 00000020
WRK_V_COMMENT                    = 0000000C
WRK_W_FLAGS                        FFFFFFF0
WRK_W_FLAGS2                       FFFFFFF2
WRK_W_IMGCHAN                      FFFFFFEE
WRK_W_PMPTLEN                      FFFFF99E
_$$_                             = 00000000

                                             +----------------+
                                             ! Psect synopsis !
                                             +----------------+

PSECT name                        Allocation           PSECT No.  Attributes
----------                        ----------           ---------  ----------
.  ABS  .                         00000000  (     0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                             FFFFFFFC  (     0.)  01 (  1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
DCL$ZCODE                         00000531  ( 1329.)  02 (  2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                          +------------------------+
                                          ! Performance indicators !
                                          +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        13    00:00:00.04    00:00:01.81
Command processing                   101    00:00:00.71    00:00:10.05
Pass 1                               238    00:00:08.66    00:00:30.92
Symbol table sort                      0    00:00:00.98    00:00:02.72
Pass 2                               196    00:00:02.55    00:00:09.47
Symbol table output                   30    00:00:00.19    00:00:00.85
Psect synopsis output                  1    00:00:00.02    00:00:00.02
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 579    00:00:13.15    00:00:55.84

The working set limit was 1200 pages.
44122 bytes (87 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 509 non-local and 146 local symbols.
1004 source lines were read in Pass 1, producing 20 object records in Pass 2.
29 pages of virtual memory were used to define 16 macros.
```

```
                                        +--------------------------+
                                        ! Macro library statistics !
                                        +--------------------------+

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                    0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                         6
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                         0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                      3
TOTALS (all libraries)                                 9

513 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:EXPRESS/OBJ=OBJ$:EXPRESS MSRC$:EXPRESS/UPDATE=(ENH$:EXPRESS)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0070 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

GETKEYNAM
LIS

GOTO
LIS

EXPRESS
LIS

HANDLE
LIS

IMAGECTRL
LIS

INITIAL
LIS

INDIRECT
LIS

FILECMDS
LIS

IF
LIS

IMAGEEXECT
LIS